A Developmental Evolutionary Learning Framework for Robotic Chinese Stroke Writing

Wu, Ruiqi; Chao, Fei; Zhou, Changle ; Huang, Yuxuan; Yang, Longzhi ; Lin, Chih-Min; Chang, Xiang; Shen, Qiang; Shang, Changjing

Published in:
IEEE Transactions on Cognitive and Developmental Systems

DOI:
10.1109/TCDS.2021.3098229

Publication date:
2022

Citation for published version (APA):
Wu, R., Chao, F., Zhou, C., Huang, Y., Yang, L., Lin, C-M., Chang, X., Shen, Q., & Shang, C. (2022). A Developmental Evolutionary Learning Framework for Robotic Chinese Stroke Writing. *IEEE Transactions on Cognitive and Developmental Systems*, *14*(3), 1155-1169. https://doi.org/10.1109/TCDS.2021.3098229

A Developmental Evolutionary Learning Framework for Robotic Chinese Stroke Writing

Ruiqi Wu, Fei Chao, *Member, IEEE,* Changle Zhou, Yuxuan Huang, Longzhi Yang, *Senior Member, IEEE,* Chih-Min Lin, *Fellow, IEEE,* Xiang Chang, Qiang Shen, and Changjing Shang

***Abstract*—The ability of robots to write Chinese strokes, which is recognized as a sophisticated task, involves complicated kinematic control algorithms. The conventional approaches for robotic writing of Chinese strokes often suffer from limited font generation methods, which limits the ability of robots to perform high-quality writing. This paper instead proposes a developmental evolutionary learning framework that enables a robot to learn to write fundamental Chinese strokes. The framework first considers the learning process of robotic writing as an evolutionary easy-to-difficult procedure. Then, a developmental learning mechanism called "Lift-constraint, act and saturate" that stems from developmental robotics is used to determine how the robot learns tasks ranging from simple to difficult by building on the learning results from the easy tasks. The developmental constraints, which include altitude adjustments, number of mutation points, and stroke trajectory points, determine the learning complexity of robot writing. The developmental algorithm divides the evolutionary procedure into three developmental learning stages. In each stage, the stroke trajectory points gradually increase, while the number of mutation points and adjustment altitudes gradually decrease, allowing the learning difficulties involved in these three stages to be categorized as easy, medium, and difficult. Our robot starts with an easy learning task and then gradually progresses to the medium and difficult tasks. Under various developmental constraint setups in each stage, the robot applies an evolutionary algorithm to handle the basic shapes of the Chinese strokes and eventually acquires the ability to write with good quality. The experimental results demonstrate that the proposed framework allows a calligraphic robot to gradually learn to write five fundamental Chinese strokes and also reveal a developmental pattern similar to that of humans. Compared to an evolutionary algorithm without the developmental mechanism, the proposed framework achieves good writing quality more rapidly.**

***Index Terms*—Developmental learning, robotic writing, evolutionary robotics, evolutionary algorithm.**

R. Wu is with the Department of Artificial Intelligence, School of Artificial Intelligence and Big Data, Henan University of Technology, China (e-mail: rqwu@haut.edu.cn).

F. Chao and C. Zhou are with the Department of Artificial Intelligence, School of Informatics, Xiamen University, China (e-mail: dozero@xmu.edu.cn; fchao@xmu.edu.cn).

Y. Huang is with the Software Development Center, Bank of Communications, China, (email: 976443263@qq.com).

L. Yang is with the Department of Computer and Information Sciences, Northumbria University, UK. (email: longzhi.yang@northumbria.ac.uk).

C.-M. Lin is with the Department of Electrical Engineering, Yuan Ze University, Taiwan. (email: cml@saturn.yzu.edu.tw).

F. Chao, X. Chang, Q. Shen, and C. Shang are with the Department of Computer Science, Institute of Mathematics, Physics and Computer Science, Aberystwyth University, SY23 3DB, UK. (email: fec10@aber.ac.uk; xic9@aber.ac.uk; qqs@aber.ac.uk; cns@aber.ac.uk).

This work was supported by the National Natural Science Foundation of China (No. 61673322, 61673326, and 91746103), Fundamental Research Funds for the Central Universities (No. 20720190142), and the Key Project of National Key R & D Project (No. 2017YFC1703303). (Corresponding author: Fei Chao.)

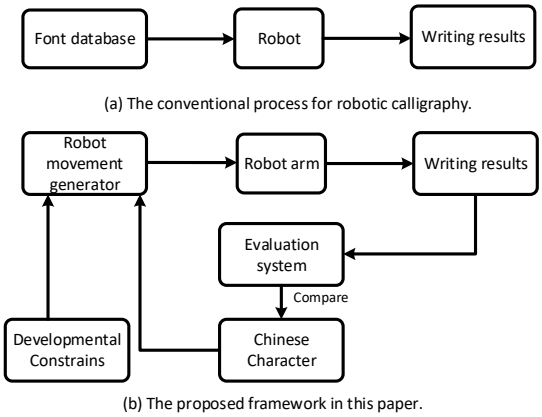


Fig. 1. (a) The main process of conventional calligraphy robots and (b) the proposed approach for the calligraphy robot.

I. Introduction

Robotic Chinese writing ability is a challenging task involving complicated control algorithms that control robotic end-effectors to write complex Chinese characters [1]–[3]. Robot Calligraphy is an important way to promote the inheritance of human culture and civilization. Writing Chinese characters is different from writing English letters because Chinese writing also considers the spatial collocations of character strokes [4], [5]. Thus, the writing quality of the character strokes is an important factor affecting the overall writing performance of Chinese characters. Excellent calligraphic robots must learn to write stroke trajectories in an intuitive, quick, and easy way. More importantly, stroke trajectory writing must be in line with human aesthetics. Therefore, imitation learning [6] and other forms of learning [7] can be used to learn from human writing and to improve on initial (preprogrammed) trajectories. If a robot can produce strokes that meet human aesthetic standards, it would also be promising for solving other human aesthetics-related tasks, such as robotic drawing and graffiti [8], [9].

The major challenge in conventional robotic Chinese calligraphy is that human engineers must design the robot stroke writing ability rather than the robots themselves autonomously developing such an ability. Therefore, the working procedure for a robot's conventional writing belongs to the "open-loop" process (see Fig. 1-a). Such a process does not contain a mechanism for the robot to perform self-evaluation and acquire feedback from the written results. Thus, in the robot

system, the robot itself cannot improve the writing results. For example, in recent research, human-computer interactive technology was used to achieve writing ability for robot manipulators [10]–[12]. In their approaches, the robots learned Chinese calligraphy from human demonstrations or through interactions with humans. To improve the efficiency of the human-robot interaction-based methods, several researchers have used human bioelectricity signals to assist with inputting human guidance information [13]. For example, Yang et al. [14] used EMG signals to remotely guide the robot to learn writing. Other studies have embedded image processing methods, stroke trajectory fitting algorithms, or predefined font databases into calligraphic robots [15]–[18]. However, without self-evaluation and feedback capabilities, robots have difficulty using these methods to improve their writing quality. In summary, due to the lack of robotic self-evaluation and feedback capabilities, writing results evaluation tasks must be conducted by humans, which increases the human workload and reduces the learning efficiency.

Another major challenge in implementing self-development or the autonomic learning capabilities of conventional calligraphic robots is the high computational cost. Mueller et al. proposed [17] a closed-loop approach using the B-spline curve-fitting algorithm for robotic calligraphy. This work required performing many learning iterations to find the optimal combinations of the applied B-spline curve equation for each type of stroke. Sasaki et al. designed a deep learning neural network that required 15,000 iterations to build relationships between their experimental robot's joints and the writing results [8]. In contrast to the curve-fitting approaches, many evolutionary algorithms have been applied to design robot controllers [19]–[21]. Recently, several deep neural network based methods to robotic writing were proposed [22], [23]. However, robotic controllers using evolutionary algorithms and neural networks also require large numbers of iterations for optimization, which is computationally intensive. Therefore, neither the curve-fitting approaches nor the evolutionary algorithm-based approaches form a quick approach by which robots can learn calligraphy.

This paper proposes an alternative developmental evolutionary framework that allows a robot to learn to write fundamental Chinese strokes with a low computational cost. As a reference model, this framework is based on the basic approach that human calligraphers take when they begin learning to write Chinese characters. A human beginner practices writing by referring to copybooks; after writing each stroke, the learner compares the writing results with the copybook templates to discover which aspects of the writing require improvement. As shown in Fig. 1, an evolutionary computational approach is used to simulate these practice-writing iterations; And, a system for evaluating the writing results is integrated into the approach. Moreover, a developmental learning algorithm named "Lift-constraint, act and saturate (LCAS)" [24] was introduced into the evolutionary approach to reduce the learning difficulty and computational cost. In the framework, the robot gradually learns to optimize its calligraphic writing. This framework extends the stroke generation ability proposed by [15]–[18] and improves the learning performance of the methods reported by [25]–[27].

The main contributions of this work are as follows:

1) A robotic developmental learning framework is established by integrating the developmental learning method with an evolutionary algorithm (in Section III-A).
2) The LCAS learning algorithm is adopted to separate a continuous optimization process into different developmental stages (in Section III-B1) by applying various types of developmental constraints to the robot. These developmental stages reduce the complexity involved in learning robotic calligraphy.

The remainder of this paper is organized as follows: Section II introduces the LCAS algorithm. Section III specifies the main implementation issues of the proposed approach. Section IV presents the experimental results and discusses their implications. Section V concludes the paper and outlines our future research.

II. Background Knowledge

Recent studies in developmental robotics reveal that developmental learning algorithms effectively reduce learning complexity as robots attempt to build highly autonomous capabilities [28]–[30]. Developmental learning in robotics has been implemented in many ways; however, from a developmental psychology perspective, lifting these constraints can lead robots to progress from a certain competence level to a new and more complex competence level [31].

From the above perspective, the LCAS algorithm first determines all the constraints that limit the robot and designs a lifting-constraint sequence to relax these constraints. Here, the constraints of developmental robotics consist of a combination of anatomical constraints (such as structural joints), sensory constraints (such as visual resolution), computational constraints (such as the capacities of neural networks), and environmental constraints (such as disturbances in a workspace) [24]. The lifting-constraint sequence defines the robot's constraints at each developmental stage.

For example, in a humanoid robot, all the robot's joints and sensors are regarded as its constraints. When the robot plans to improve its hand-eye coordination skill, the constraints on the robot's vision and arm joints are removed, but the constraints on the lower limb joints of the robot remain. Therefore, the humanoid robot can concentrate on learning robotic hand-eye coordination. After the robot has improved its hand-eye coordination ability, it can turn its attention to improving its walking ability; then, the constraints on the lower limb joints are removed and the robot concentrates on learning to walk. After the lifting-constraint sequence has been established, the next step in LCAS is for the robot to search for novel stimuli, allowing the robot to learn under fully constrained conditions. When the saturation rate of the robot learning system of a robot becomes stable, the current constraints are removed, and the next constraints in the constraint sequence are assigned to the robot. Then, the robot repeats the search for novel stimuli in an attempt to learn under this new condition. The robot will have some new abilities after it has experienced all the constrained conditions. The LCAS algorithm has been successfully applied to several developmental models [32], [33].

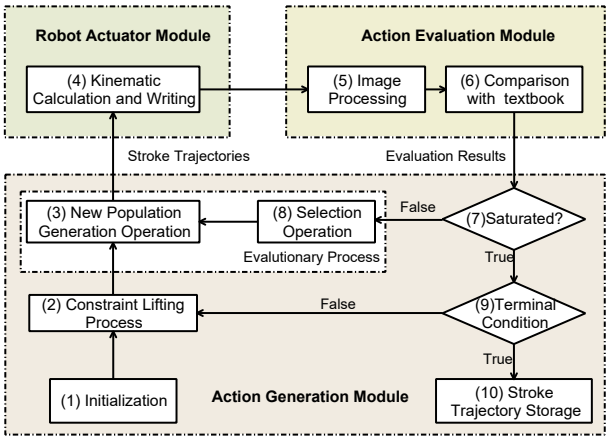


Fig. 2. The steps in the developmental evolutionary learning framework for the calligraphy robot.

Based on the LCAS algorithm, the calligraphic robot first develops a rough stroke style and then gradually generates more detailed stroke shapes. Thus, the robot can learn new writing skills without complex programming and a large amount of reiterative training to greatly reduce the complexity level of a robotic writing task. Moreover, robots imitating human developmental procedures exhibit faster learning speeds than do robots without developmental features. Therefore, this work improves robots' autonomous abilities and reduces their learning complexity. Additionally, this work introduces a new robotic learning framework by combining developmental learning and evolutionary algorithms to reveal a new approach to developmental robotics.

Algorithm 1 outlines the LCAS algorithm. In this algorithm, i denotes the current learning time under the constrained environment, and $Sat(i)$ is a function to determine whether the saturation value has met the saturation threshold during the ith learning iteration.

Algorithm 1 Procedure of the LCAS-based Approach

```
1: while constraint sequence is not null do
2:    Issuing a new constraint from the constraint sequence;
3:    i = 0
4:    while Sat(i) is FALSE do
5:       Select the best individuals from the current population;
6:       Generate a new population;
7:       Perform kinematic calculations and writing strokes;
8:       Capture and process stroke images;
9:       Calculate the similarity of each individual stroke;
10:      i++;
11:   end while
12: end while
```

III. Proposed Approach

A. Framework Procedure

This study fuses the ideas of "evolutionary computation" and "developmental learning"; therefore, our robot continuously optimizes the results of writing Chinese strokes through autonomous learning, just as beginning human calligraphers learn to produce Chinese calligraphy. Fig. 2 illustrates the robot's developmental evolutionary learning framework for generating autonomous Chinese strokes.

The framework is composed of three modules: 1) Action Generation, 2) Robot Actuator, and 3) Action Evaluation. The Action Generation module performs trajectory generation based on a developmental evolutionary framework that combines evolutionary computation and developmental learning algorithms. In this module, the developmental learning algorithm divides the learning process of the robot into three stages; then, the robot learns based on an evolutionary algorithm executed in each stage. The Robot Actuator module performs kinematic calculations for the robot and controls its actions. The Action Evaluation module captures the results of the robot's writing and evaluates the quality of each trajectory.

The elements in the unshaded area of the Action Generation module constitute a typical evolutionary algorithm; the elements in the shaded area determine the developmental stages during the evolutionary algorithm. The main procedure is as follows: The LCAS algorithm divides the evolutionary computation into three developmental stages and builds a constraint-lifting sequence (Step 1). The constraint sequence includes the constraints at each stage. Then, the constraint of the first stage is lifted from the robot (from Step 1 to Step 2). Following constraint lifting (Step 2), stroke trajectories are generated by the new population generation system (Step 3). Next, the trajectory coordinate parameters of these stroke samples are transferred to the Robot Actuator module (from Step 3 to Step 4), which applies kinematic calculations to convert the stroke trajectories to the manipulator's motor commands, causing the robot to write each stroke on the writing board (Step 4).

In the Action Evaluation module, the writing results are captured by an RGB camera mounted on the manipulator's gripper (Steps 4–5). After the image is processed (Step 5), the written stroke is compared with its related stroke in the calligraphy textbook to obtain the similarities (Step 6). The similarities, which are regarded as the evaluation results are sent back to the Action Generation module (Steps 6–7). The developmental learning algorithm, "LCAS", then determines whether to move to the next developmental stage (Step 7). If not, the Action Generation module uses the conventional evolutionary procedure to select high-performing individuals from the previous population (Step 8) and generates a new population for the next iteration (moving from Step 8 to Step 3). If LCAS decides to move to the next stage, the module checks whether the termination criteria have been reached (Step 9). If so, the stroke trajectory with the best writing quality is retained in the Stroke Trajectory Storage, and the iteration stops (Step 10). If not, the LCAS algorithm switches from the current developmental stage into the next stage. Within each stage, the parameters of the evolutionary computation differ from those of the other stages (from Step 9 to Step 2). The termination criterion is reached when all the constraints on the constraint sequence have been lifted.

The following subsections describe the main steps in the proposed framework.

TABLE I
PARAMETERS AND CORRESPONDING SYMBOLS USED IN THE LEARNING PROCESS

Parameters	Symbols
Iteration No.	t
Trajectory Point No.	N
Mutation Point No.	μ
Point Mutation Amplitude	α
Population Size	Q
Saturation Threshold 1	ε
Saturation Threshold 2	ψ

B. Action Generation Module

The Action Generation module first initializes the parameters for the evolutionary computation and LCAS. The evolutionary computation parameters include 1) the initial population, 2) the size of the population (number of individuals), 3) the maximum number of iterations, and 4) the mutation amplitude. The LCAS parameters include the saturation thresholds for Stages 1 and 2. The parameters and their corresponding symbols are given in Table I.

The Action Generation module includes two main processes: 1) the constraint-lifting process and 2) the evolutionary process. The constraint-lifting process defines the developmental stages of the learning framework. The evolutionary process applies a regular evolutionary programming algorithm (EP) [34] to optimize the stroke trajectories within each developmental stage. The evolutionary process is composed of a new population generation operation and an individual selection operation.

1) Constraint-lifting Process: The entire development process consists of a sequence of developmental stages. The learning complexity increases from the first to the final stage. In other words, the robot learns the easiest tasks during the first stage. After the robot's learning results have become stable, the robot moves to the second stage and starts to learn more complicated tasks. When the robot's writing performance again becomes stable, the robot moves to the last stage to learn the most complicated tasks. The developmental process is defined as follows:

1) First, the robot must "act" under the constraints of the current stage. In this work, "act" refers to the evolutionary process.
2) Then, the robot checks whether its learning has been "saturated". Here, "Saturated" means that the robot's writing performance is stable.
3) If it is "Saturated", the robot moves to the next developmental stage, and new "constraints" are applied to the robot; thus, the robot executes Step 1.
4) If it is not "Saturated", no new "constraints" are applied to the robot, and the robot returns to Step 1.

The measure of task complexity is based on three constraint parameters: 1) the number of trajectory points of each stroke, N; 2) the number of mutation points, μ; and 3) the mutation amplitude of the trajectory points, α. When the robot moves from one stage to the next, the values of the three constraint parameters change. During the developmental process, as N increases, the writing task becomes increasingly difficult. However, when the μ and α parameters of EP are too high, the skills learned during the previous stages can be easily lost during the optimization process. To avoid this problem, μ and α are decreased at each change of developmental stage. The implementation of the three developmental stages is described as follows:

First stage: In this stage, a random function generates an initial stroke population. To ensure the population diversity, the initial population does not include a mutation operation: all individuals are generated randomly. Each individual is composed of N stroke trajectory points, of which μ mutation points are randomly selected from N. The change values are defined according to Eq. 6. The first-stage trajectory is represented by:

$$\begin{aligned} T_s^{s1} :&[(x_1, y_1, z_1), (x_2, y_2, z_2), \cdots, (x_N, y_5, z_N)] \\ &x, y, z \in \mathbb{R}; \quad 0 \leq x, y, z \leq 400 \end{aligned}. \tag{1}$$

Tournament selection is used to choose the individuals to participate in the next iteration of new groups during the evolution and mutation processes. Each iteration of individuals is written by the robot actuator module, in which each trajectory is converted from the Cartesian space (Eq. 1) to the joint space of a robot through the inverse kinematics equations for the robot (see Section III-C for details). In the Cartesian space, (x, y) represents the horizontal position of the writing board, and z denotes the height of the pen, so as to simulate the writing force of the pen.

The robot actuator module controls a 5-DOF manipulator whose joint space is defined as shown in Eq. 2, where p_i denotes the $i-th$ trajectory point, and j_i denotes the value of the $i-th$ joint of the robot. After the conversion, the trajectory, P_s^{s1}, is sent to the robot, which performs the trajectory motion. Then, the robot moves from p_i to p_{i+1} through continuous interpolation. The working angle range of each joint is shown in Table II.

$$\begin{cases} P_s^{s1} : [p_1, p_2, \cdots, p_5] \\ p_i : [j_1, j_2, \cdots, j_5] \end{cases}. \tag{2}$$

Next, evaluation scores $Eva(t)$ are calculated by the Action Evaluation module. According to the LCAS algorithm, a saturation function is used to determine whether the robot's learning process is stable. Here, the saturation function is expressed as follows:

$$Sat(Eva(t), t) = \begin{cases} true; & \text{if } |Eva(t) - Eva(t-\varphi)| \\ & < \psi \quad \text{and } Eva(t) > \varepsilon \\ false; & \text{else} \end{cases}, \tag{3}$$

where $Eva(t)$ represents the evaluation score calculated by the Action Evaluation module at the tth generation, and ε and ψ are specific thresholds set in advance. When the change in $Eva(t)$ is less than when the generation increment is φ and $Eva(t)$ is greater than ψ, the robot's learning at this stage is stable. The value of φ is set based on the complexity of the task. In this study, the value of φ was empirically set to

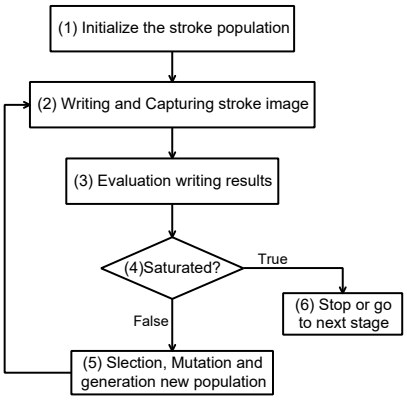


Fig. 3. The evolutionary process within each developmental stage.

1. In addition, N, μ, and α change according to the LCAS algorithm; then, the robot moves to the next stage.

Middle stages: At middle stages, the robot acts based on the new N, μ, and α values. In each stage, N is increased, and the new points are evenly inserted into the trajectory points generated in the previous stage. Here, a simple interpolation method is used to add new stroke trajectory points; thus, the position values of a new point are defined as follows:

$$\begin{cases} x_i = \dfrac{x_{i-1} + x_{i+1}}{2} \\ y_i = \dfrac{y_{i-1} + y_{i+1}}{2} \\ z_i = \dfrac{z_{i-1} + z_{i+1}}{2} \end{cases}, \tag{4}$$

where $(x_{i-1}, y_{i-1}, z_{i-1})$ and $(x_{i+1}, y_{i+1}, z_{i+1})$ denote the points $(x_{i-1}, y_{i-1}, z_{i-1})$ and $(x_i, y_i, z_i))$ in Eq. 1, respectively.

The saturation function (Eq. 3) is used to determine whether the robot's learning performance is stable; if so, the robot moves to the next developmental stage.

Final stage: After entering the final stage, the constraint sequence becomes null, and the parameters N, μ, and α reach their limit value. The saturation function (Eq. 3) is again used to determine whether the robot's learning performance is stable. When the final effect of the written stroke reaches the experimental requirement (i.e., when the final stage's learning performance is stable), both the evolutionary and developmental processes terminate. The individuals with the best writing quality are retained in the Stroke Trajectory Storage for Chinese character writing.

2) Evolutionary Process: The evolutionary process is as follows: In each stage, N stroke trajectory points representing a stroke are randomly generated. Then, μ mutation points are randomly selected from the set of trajectory points. The optimized results of the strokes are achieved through repeated iterations. As shown in Fig. 3, the specific evolutionary process within each developmental stage is as follows:

Step 1: Using the parameters for the current stage, generate some strokes consisting of N random stroke coordinate points during initialization.

Step 2: Each generated stroke is written by the robotic manipulator. Then, the writing results are captured by the image preprocessing system to obtain images of the written strokes.

Step 3: The written strokes are evaluated using the stroke evaluation method and the evaluation score is calculated for each written stroke.

Step 4: The LCAS algorithm checks whether the learning is saturated. If so, the process moves to the next development stage; or if in the last stage, the process of stroke generation is completed. The resulting stroke with the highest score will be learned and stored by the robot. If the learning is not saturated, stroke samples with poor quality are removed based on the evaluation scores. Meanwhile, several randomly selected points in the stroke trajectory points are mutated based on the number of mutation points, μ, to generate a new population. The generation method is specified in the next subsection.

Step 5: Return to Step 2.

a) New Population Generation Operation: This operation consists two of generation modes: one is used only before the first iteration of the population, in which a random function generates the first iteration of strokes, while the other mode is used for all subsequent iterations.

The system generates a new iteration based on the previous (parent) iteration's population. In the new population, the best individuals are retained from the previous population, and the rest are generated by mutations of these previous individuals. The mutation operation is based on the following mutation function:

$$\begin{cases} x(t+1) = x(t) + \Delta x(t) \\ y(t+1) = y(t) + \Delta y(t) \\ z(t+1) = z(t) + \Delta z(t) \end{cases}, \tag{5}$$

where t represents the total number of iterations (i.e., evolutionary generations), and $x(t+1)$ represents the progeny generated by adding the step size $\Delta x(t)$ to the parent. In $(\Delta x(t), \Delta y(t), \Delta z(t))$, the point value is $(0, 0, 0)$ unless it is the mutation point. The value of the mutation point is defined as follows:

$$\begin{cases} \Delta x_m = \sqrt{1 - Eva(t)} \cdot N_x(0, \alpha) \\ \Delta y_m = \sqrt{1 - Eva(t)} \cdot N_y(0, \alpha) \\ \Delta z_m = \sqrt{1 - Eva(t)} \cdot N_z(0, \alpha) \end{cases}, \tag{6}$$

where $Eva(t)$ represents the evaluation score from the Action Evaluation module whose value range is $[0, 1]$; $N_x(0, \alpha)$, $N_y(0, \alpha)$and $N_z(0, \alpha)$ are Gaussian noise functions whose mean and variance are set to 0 and $alpha$, respectively. The value of α in the three developmental stages is determined by the LCAS algorithm.

b) Selection Operation: Tournament selection (a random type of q-competition) is adopted [35] as the selection operator in this evolutionary process. First, among all the strokes Q, q strokes are randomly selected to form a group. Then, the stroke with the highest score from q is retained for mutation; the rest are returned to Q. The value of q is usually set to $0.45Q$. Because no crossover operator exists in this evolutionary programming algorithm, new generations are based on the mutation and selection operators. The number of selected individuals is set to $0.5Q$, and the selection process is repeated

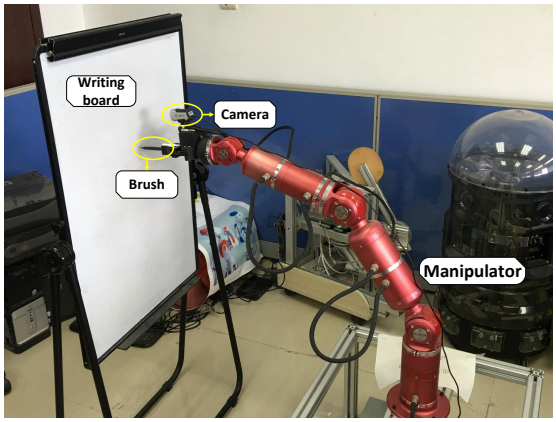


Fig. 4. The experimental robot system consisted of a 5-DOF manipulator, a calligraphy brush, and an RGB camera.

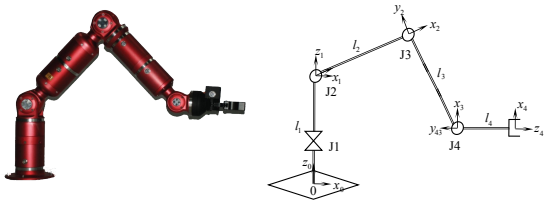


Fig. 5. The kinematic configurations of the experimental manipulator.

until $0.5Q$ strokes have been selected. Then, these selected strokes and their mutated offspring form the new population. For the selection operation, the values of Q and q are set to 20 and 9, respectively.

C. Robot Actuator Module

Fig. 4 depicts the experimental robot system, which consists of a 5-DOF manipulator, a calligraphy brush held by the manipulator's gripper, and an RGB camera mounted on the gripper. A white writing board is placed vertically within the manipulator's working range. The positions of both the manipulator and the writing board are fixed. After writing a stroke, the manipulator returns to a predefined position; then, the RGB camera captures the written stroke and delivers the resulting images to the robot evaluation system, which consists of a control computer and a hardware controller.

The robot system is connected to the hardware controller, which is responsible for low-level motor control, via the CAN bus. The RGB camera is connected to the control computer via USB. The controllers for both the motor and the sensor systems are installed on the AVR controller. A program that controls an RSC-232 socket and integrates the controller driver programs were built to communicate between the control computer and the hardware controller.

Because the calligraphy brush is fixed on the gripper, the gripper joint is set to a fixed value in the writing task. Fig. 5 shows the kinematic configuration of the experimental manipulator. The configuration includes the coordinate frame of each joint and the setup of the joints and links of the manipulator. Within the Cartesian space defined by x_0, y_0, and z_0, the experimental manipulator can move from the center (origin) to any other location. The joints of the experimental manipulator are denoted by j_1, j_2, j_3, and j_4, and the links are denoted by l_1, l_2, l_3, and l_4; the links have lengths of $150mm$, $375mm$, $354mm$, and $175mm$, respectively, where l_1 is the distance from the horizontal plane to j_2, l_2 is the distance between j_2 and j_3, l_3 is the distance between j_3 and j_4, and l_4 is the distance between j_4 and the gripper.

TABLE II
D-H PARAMETER TABLE OF THE ROBOT.

Joint index	Joint angle θ	Link offset d	Link length a	Link twist α	Working angle range
1	θ_1	0	150	-90	$[-120°, 120°]$
2	θ_2	0	375	0	$[-90°, 90°]$
3	θ_3	0	354	0	$[-90°, 90°]$
4	θ_4	0	175	0	$[-45°, 45°]$

The Denavit and Hartenberg (D-H) convention is used to analyze the forward kinematics of the robot. The D-H parameters are listed in Table II. According to the D-H parameters and the Cartesian coordinate of the robot (p_x, p_y, p_z), the joint value of the robot can be calculated by the following equations:

$$\theta_1 = \arctan \frac{p_x}{p_y}, \tag{7}$$

$$\theta_2 = \arcsin - \left(\frac{a_2 p_z + a_3 \cos\theta_3 p_z}{{a_2}^2 + {a_3}^2 + 2a_2 a_3 \cos\theta_3} + \frac{a_3 \sin\theta_3 \sqrt{{a_2}^2 + {a_3}^2 + 2a_2 a_3 \cos\theta_3 - {p_z}^2}}{{a_2}^2 + {a_3}^2 + 2a_2 a_3 \cos\theta_3} \right), \tag{8}$$

$$\theta_2 = \arccos \frac{(p_x \cos\theta_1 + p_y \sin\theta_1 - a_4)^2 + {p_z}^2 + {a_2}^2 - {a_3}^2}{2a_2 a_3} \tag{9}$$

$$\theta_1 = -\theta_2 - \theta_3. \tag{10}$$

D. Action Evaluation Module

The Action Evaluation module is used to assess the writing effect of each stroke to gradually improve the robot's writing ability. The evaluation score is fed back to the Action Generation module.

Simple methods similar to RMSE cannot well calculate similarities between written strokes and captured strokes from textbooks. Such similarities however are interfered with by many factors, such as different light intensities, focal lengths, and offsets. These simple methods cannot eliminate the interferences of these factors. In contrast, the wavelet transform has been proven effective for computing the similarity of two Chinese characters [36]. This evaluation method was developed in our previous work [37]. The computational cost of this evaluation model is low compared with the cosine similarity and the evaluation method that computes pixel-space distances directly [37]. The Action Evaluation module consists of three components: 1) image signal projection, 2) signal wavelet transform part, and 3) result analysis and evaluation.

1) Image Signal Projection: Before extracting the projection feature, the stroke image must be preprocessed. The stroke image after image preprocessing (binarization) is expressed as follows:

$$Img(i,j) = \begin{cases} 0, & (i,j) \notin Stroke \\ 1, & (i,j) \in Stroke \end{cases}, \tag{11}$$

where i and j denote the pixel index of each stroke image. The pixels that form part of the stroke are defined as one, while pixels that are not part of the stroke are defined as zero. After preprocessing, the vertical and horizontal projections of the image can be calculated. First, the pixels in each row or column of the binarized image are summed, converting the image into two histograms. The outlines of these two histograms are the vertical and horizontal projections of the image signal.

2) Signal Wavelet Transform: The compacted orthogonal wavelet Daubechies wavelet function (i.e., the db4 wavelet function with vanishing moments of 4) is used and evaluated by four scales. On the m scale ($m = 1, 2, 3, 4$), the sample horizontal coefficient is defined as Q_{im}, the sample vertical coefficient is defined as Q_{jm}, the standard horizontal coefficient is defined as Q_{sim}, and the standard vertical coefficient is defined as Q_{sjm}. Note that the sample coefficients come from the robot writing image, and the standard coefficients come from the stroke image of the textbook. A larger difference between a sample coefficient and a standard coefficient indicates poorer robot writing quality. The differences between the horizontal direction coefficient and the vertical direction coefficient are expressed as follows:

$$D_{im} = \sum_{k=1}^{4} |Q_{im}(k) - Q_{sim}(k)| \tag{12}$$

$$D_{jm} = \sum_{k=1}^{4} |Q_{jm}(k) - Q_{sjm}(k)|, \tag{13}$$

where k represents the corresponding scale value (abscissa value) between the stroke sample and the textbook's stroke. This is equivalent to visually using four different distances to observe the written stroke (the distance of scale 1 is closest, the distance of scale 4 is farthest). Thus, we obtain four different values.

3) Result Evaluation: The difference of the high-frequency coefficients of four different scales is processed by a fuzzy reasoning system to calculate the final evaluation scores for each stroke. In this system, the maximum score is 100 and the final evaluation score is a real number from 0 to 100.

The means of the differences in the coefficients of the four scales are calculated. Thus, for the vertical and horizontal values, D_i and D_j, we have:

$$D_i = \frac{D_{i1} + D_{i2} + D_{i3} + D_{i4}}{4}.\ D_i \in \mathbb{R};\ 0 \le D_i \le 100, \tag{14}$$

and

$$D_j = \frac{D_{j1} + D_{j2} + D_{j3} + D_{j4}}{4}.\ D_j \in \mathbb{R};\ 0 \le D_j \le 100, \tag{15}$$

TABLE III
PARAMETER SETTINGS FOR EACH STAGE IN LCAS

Constraint Type	Stage 1	Stage 2	Stage 3
Trajectory Point No. N	5	7	10
Mutation Point No. μ	4	3	2
Point Mutation Amplitude α	30	20	15
Saturation Threshold 1 ε	0.4	0.6	0.8
Saturation Threshold 2 ψ	0.02		

respectively. Because coefficient differences exist in both the row and column directions, the ratio value adopted here is 50%. Then, before calculating the evaluation score, the stroke evaluation scores in the horizontal and vertical directions are scaled to real numbers from 0 to 50. The scaling formula is then defined as follows:

$$D(X) = \begin{cases} D_i(X),\ if\ D_i(x) < 50; \\ 50,\ else; \end{cases} \tag{16}$$

$$D(Y) = \begin{cases} D_i(y),\ if\ D_i(y) < 50; \\ 50,\ else; \end{cases} \tag{17}$$

Finally, the evaluation score is obtained by

$$\begin{aligned} Eva =& 0.01 \cdot (100 - D(X) - D(Y)). \\ & D(X), D(Y) \in \mathbb{R};\ 0 \le D(X), D(Y) \le 50 \end{aligned} \tag{18}$$

The maximum value and minimum values of $D(X)$ and $D(Y)$ are 50 and 0, respectively. In addition, the function of the parameter 0.01 in Eq. 18 is to scale Eva to a range of $[0, 1]$. Thus, the final evaluation score is a real number with a range between 0 and 1, which is invoked by the Action Generation module.

IV. EXPERIMENTS

The experiments involved two evaluations: (1) the process for learning to write five types of Chinese character strokes and (2) a comparison between the LCAS method and the evolutionary programming method without any added features. In addition, a systematic analysis of the appearance of the phenomenon during the process of autonomous stroke generation was conducted, and the experimental results are described in this section.

A. Experimental Setups

During the process of autonomous stroke generation, the LCAS rules are formulated according to Table III. Note that the threshold parameters of the saturation function (Eq. 3) are set empirically. The saturation thresholds ε in Stages 1, 2, and 3 were set to 0.4, 0.6, and 0.8, respectively. The other two saturation thresholds ψ and φ were set to 0.02 and 1, respectively, in all three stages. The remaining parameters of the EP were fixed in all the developmental stages. A too-large population increases the difficulty of the robot for generating many strokes. However, when the population is too small, the robot will quickly lose variations. To balance the generation speed of the robot and the exploration quality of the algorithm,

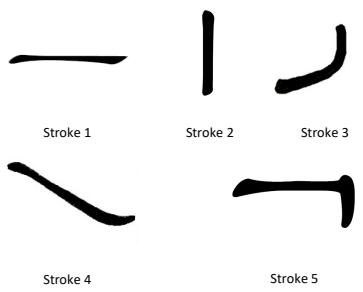


Fig. 6. The five standard strokes in the "Li" style.

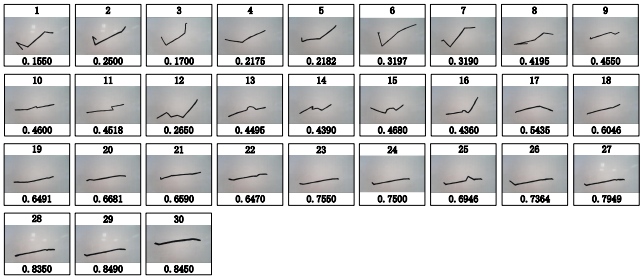


Fig. 7. Learning history of the "Horizontal" stroke.

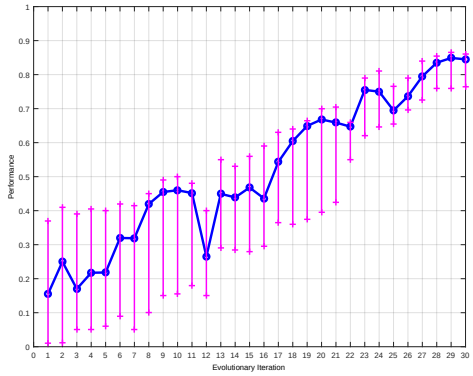


Fig. 8. Evaluation scores during the entire learning process of the "Horizontal" stroke

the value of Q was empirically set to 20. Thus, the number of selected individuals was set to 10; and the number of competitors, q, was set to 9.

In addition, the writing force is an important factor for calligraphy art. In the revision, we have added the writing force to the proposed learning framework. In this version, the z-axis information of each trajectory point (x,y,z) is used to simulate the writing force. However, in the experimentation, limited by the control precision of the robot and the material of the brush, the writing height was set to a fixed value.

B. Learning Process of Stroke Autonomous Generation

The five types of Chinese character strokes in Fig. 6 are selected from two "Li style copybooks" that are used as standard writing templates from which the robot learns. Writing Chinese characters by robots is suitable for evaluating the effectiveness of a robot learning framework. Figs. from 7 to 16 show the learning processes of our robot to write these five strokes.

Fig. 7 shows the learning history for the "Horizontal" stroke, involving a total of thirty generations. Each subfigure, selected from each generation, consists of three components: 1) the generation index, 2) the writing result, and 3) the average evaluation score. The generation number for each subfigure is shown in the label at the top of each subfigure. The writing results shown in the middle of the subfigure are images captured from the writing board after the robot has completed a writing stroke. The average evaluation score (shown in the label at the bottom of each subfigure), is based on the corresponding writing result and calculated by the Action Evaluation module. For this figure, we selected only the writing results whose evaluation scores were the highest in each generation.

In Fig. 7, the quality of the writing results gradually improves. In the beginning, the shapes of the writing results are totally different from those in the template. However, the evolutionary process guides the robot to ignore poor-quality strokes and use higher-quality strokes to generate new strokes. In particular, the results from the twenty-seventh to thirtieth generations achieve a high level. By the twelfth generation, the LCAS algorithm causes the average evaluation score to decrease significantly. The LCAS algorithm, by placing new constraints on the robot, causes the robot to act in a new environment. The constraints create a larger solution space for the robot; however, after applying the constraints, the larger solution space also reduces the overall performance during the first few generations. This decrease causes the robot to re-learn the writing task.

Fig. 8 shows the evaluation scores during the entire learning process of the "Horizontal" stroke. The solid points indicate the average evaluation scores of all the stroke samples in each generation. The range of the evaluation scores in each generation is indicated by a solid line between the minimum and maximum values.

During the stroke autonomous learning process, the three stages of LCAS are clearly apparent. The saturation state is first achieved in the eleventh generation. At this point, the first stage ends, and the constraints are replaced; consequently, starting in the twelfth generation, the scores decline significantly. Then, through mutation and evolution, the evaluation scores of the written stroke gradually improve again. The second saturation state is attained in the twenty-first generation. This time, after the constraints are replaced, the score shows only a slight decrease compared with the score decrease in the twelfth generation. This smaller decrease occurs because although more trajectory points have been added to the robot, both the number of mutation range points and the mutation range are reduced. After mutation and evolution, the learning process for the "Horizontal" stroke is completed in the thirtieth generation.

Fig. 9 shows the learning history of the "Vertical" stroke. The arrangement of the subfigures in this figure is identical to that of Fig. 7. However, the learning requires a total of 35 generations. The learning history of the "vertical" stroke is similar to that of the "horizontal" stroke. The last two

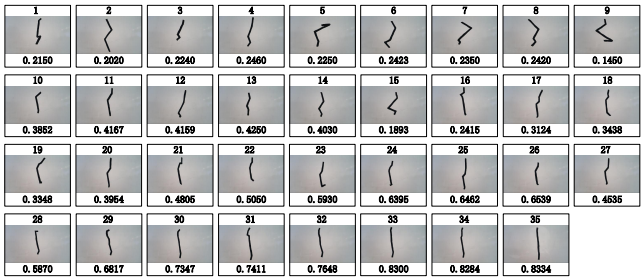


Fig. 9. Learning history of the "vertical" stroke.

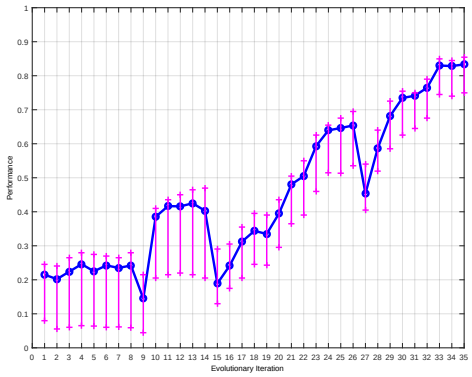


Fig. 10. Evaluation scores during the entire learning process of the "Vertical" stroke

constraint replacements occur in the thirteenth and twenty-sixth generations, respectively. When the average evaluation score reaches 0.8334, the final learning process terminates.

Fig. 10 shows the evaluation scores during the entire learning process of the "Vertical" stroke. In particular, after the fifteenth generation, the score decreases from approximately 0.4 to 0.2. This sharp decrease is caused by the new constraints. Then, the average scores increase slowly until the twenty-sixth generation, where the constraints are replaced for the second time. Again, the average scores decrease, but at a lesser magnitude than that of the first replacement. Finally, the learning process of the "Vertical" stroke is completed in the thirty-fifth generation.

Fig. 11 shows the learning history of the "Left Falling" stroke. The "left falling" stroke (a curved trajectory) has a more complicated shape than do the "horizontal" and "vertical" strokes. Therefore, in this case, the total number of learning generations required by the robot increases to 39. However, no performance decrease after replacement can be readily identified from this figure.

Fig. 12 shows the evaluation scores during the entire learning process of the "Left Falling" stroke. Saturation is achieved for the first time in the fourteenth generation. In the fifteenth generation, the average score falls from approximately 0.4 to 0.3. The second saturation stage is reached in the twenty-seventh generation, at which point the score decreases again, this time with a magnitude is greater than that after the first replacement. However, the score recovers rapidly.

Fig. 13 shows the learning process of the "Right Falling" stroke. The learning process includes a total of 28

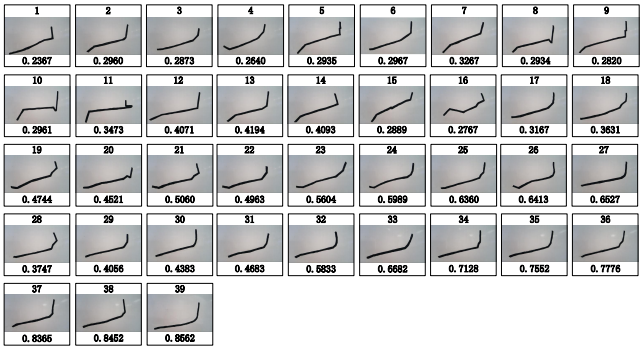


Fig. 11. Learning history of the "Left Falling" stroke.

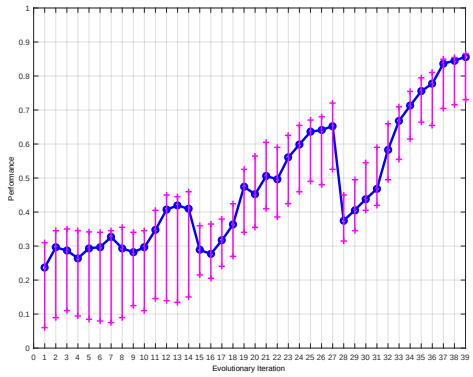


Fig. 12. Evaluation scores during the entire learning process of the "Left Falling" stroke

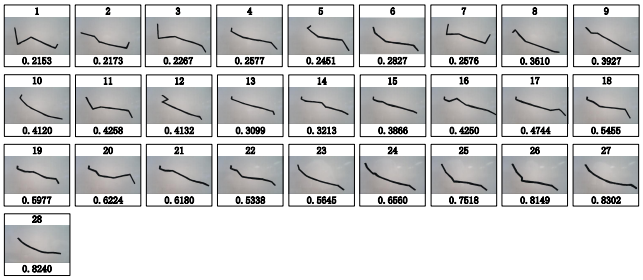


Fig. 13. Learning history of the "Right Falling" stroke.

generations—the fewest of the five strokes. This stroke also involves a curved trajectory; however, its radius is slight. Thus, the shape of the "Right Falling" stroke is quite similar to that of the "horizontal" stroke.

Fig. 14 shows evaluation scores during the entire learning process of the "Right Falling" stroke. Constraint replacements occur at the twelfth and twenty-first generations. The magnitudes of the performance decreases in both cases are slight compared to the performance decreases found with the other strokes. This small decrease leads to the fewest number of generations. The learning process of this stroke is completed at the twenty-eighth generation.

Fig. 15 shows the learning history of the "Fold" stroke. Of the five strokes, this stroke is the most complicated and its learning history shows that it requires 43 generations. This stroke is similar to a combination of the "horizontal" and "vertical" strokes, and its complicated shape requires more

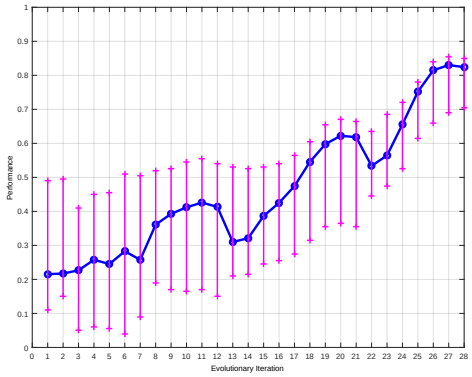


Fig. 14. Evaluation scores during the entire learning process of the "Right Falling" stroke

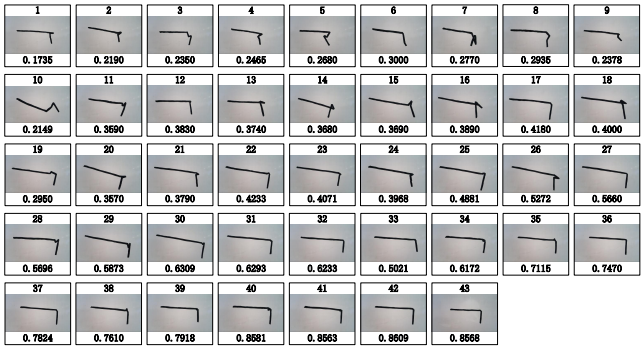


Fig. 15. Learning history of the "Fold" stroke.

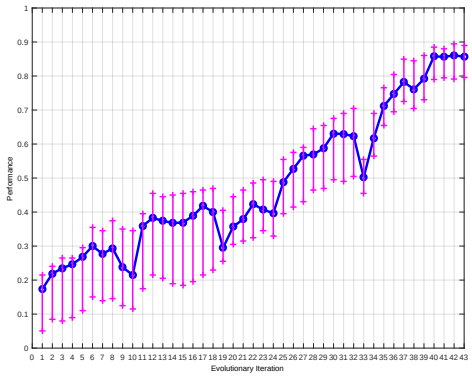


Fig. 16. Evaluation scores during the entire learning process of the "Fold" stroke

generations for the robot to learn. More than three significant performance decreases are shown in this figure; two are caused by the LCAS algorithm. Note that Figs. 7, 9, 11, 13, and 15 merely illustrate the writing results with the highest score. If a result with a trajectory crossing situation is assessed, the result might not reflect the highest scores; therefore, these figures do not include many trajectory crossing situations.

Fig. 16 shows the evaluation scores during the entire learning process of the "Fold" stroke. As mentioned in Fig. 15, the entire curve fluctuates violently. The saturation state is achieved for the first time at the eighteenth generation. In the nineteenth generation and subsequently, large reductions occur in the score. By the thirty-second generation, the score

TABLE IV
THE END GENERATION AND AVERAGE SCORE OF EVERY STAGE OF THE FIVE STROKES.

Strokes	Stage 1		Stage 2		Stage 3	
	$i(t)$	score	$i(t)$	score	$i(t)$	score
Stroke 1	11	0.4518	21	0.6590	30	0.8450
Stroke 2	13	0.4250	26	0.6539	35	0.8334
Stroke 3	14	0.4093	27	0.6527	39	0.8562
Stroke 4	12	0.4132	21	0.6180	28	0.8240
Stroke 5	18	0.4000	32	0.6233	43	0.8568

decreases substantially again. Then, through mutation and evolution, the learning process of this stroke is completed in the forty-third generation.

Generally, the learning results of these five strokes proved that the learning framework was effective for robot learning calligraphy tasks. Moreover, considering the autonomous generation process of these five kinds of strokes, we have two findings. (1) The number of generations required for different stroke types is not identical. Among the tested strokes, the number of evolutionary generations of "Horizontal", "Vertical", "Left Falling" and "Right Falling" were all approximately 30; however, the "Fold" stroke required more than 40 generations to reach the experimental requirements. (2) As the generations evolve, the changes between the evaluation scores' minimum and maximum values move from large to small. The change in this range is much larger at the beginning of the evolution; however, the range changes decrease for all five strokes. Table IV lists the number of generations required to reach saturation in each stage for all the strokes.

C. Comparative Analysis of Learning Processes

Fig. 17 shows the writing results of the five strokes when applying the evolutionary method with the developmental algorithm, LCAS (solid blue line), and the evolutionary programming method without the algorithm, EP (dotted red line) to the autonomous generation process. The parameters used for EP are those of the third stage in Table III. Before the termination of each stroke, the maximum number of generations for EP was set to match the number of generations for the same stroke required by LCAS. For example, in LCAS, the "horizontal" strokes terminate at the thirtieth generation; thus, in EP, the "horizontal" stroke maximum number of generations was set to thirty.

In the generation processes of the five strokes, although the evaluation scores still rise generation by generation, the evaluation scores of the EP method are generally lower than those of the LCAS method. At the beginning of the evolutionary process, the score differences between the two methods are not obvious; however, the final scores of EP are much lower than those of LCAS. The EP method requires more evolutionary generations to achieve scores equivalent to those of LCAS. This situation indicates that the learning rate of the proposed developmental evolutionary approach is more efficient than that of the conventional evolutionary method.

As shown in Fig. 17e, in EP, the performance curve of the "Fold" stroke is not as smooth as are the curves of the other

strokes. In the "Horizontal", "Vertical", "Right Falling", and "Left Falling" strokes, the EP curves have steadily increasing trends. In addition, the two curves are intertwined until after the thirty-third generation; then, the LCAS scores increase substantially, mainly because the complexity of writing the "Fold" stroke is the highest among the five strokes. This complexity leads to unstable performances for both EP and LCAS. Furthermore, in the EP method, ten trajectory points are used from the beginning of the evolutionary process; in contrast, the LCAS method has ten trajectory points only in the third stage (the number of coordinates in the first and second stages are 5 and 7, respectively). Therefore, the large increase in LCAS in the thirty-fourth generation benefits from more trajectory points, helping to support the complicated shape of the "Fold" stroke.

D. Qualitatively Comparison

To further reveal the strengths of the proposed model, we compared the proposed method with our two previous work and two work of other researchers. Due to these work has different research purposes and methods, we can only use the qualitative comparison method to compare them.

In the first comparison work [5], the robot learns to write base strokes by using human gestures. First, a Chinese character is decomposed to a set of base strokes. Then, these decomposed strokes are matched with the writing trajectory stored in a database. finally, the robot uses the stroke trajectories to write Chinese characters. The second comparison work [38] is another study from us. The robot learns to write simple strokes and English letters via human arm gestures; then, it uses the learned knowledge and human gestures to write Chinese characters and English words. The third work, proposed by Liang et al. [39], applied a generative adversarial networks-based method to convert a character font style in the network's input image to a new font style in the output image. Then, the robot writes the character with the new style by sampling the output image. The fourth work, proposed by Zhang et al. [40], firstly recognizes characters from input images through a pre-trained neural network, and retrieves the corresponding standard strokes from a pre-built calligraphy font library. Then, the method generates robotic writing actions by using the retrieved strokes to write input characters.

Compared with the above four methods, the advantages of the work proposed in this paper can be summarized from four aspects: (1) Evaluation mechanism; (2) Robot autonomy performance; (3) Control mode; and (4) Computational cost. The comparison results are summarized in Table V. The detailed comparisons are listed as follows:

(1) Evaluation mechanism. No evaluation method is established in the first three methods; the results are evaluated only by humans. The fourth method uses the minimization differences between the handwritten strokes and standard calligraphy strokes to evaluate the generated stroke. In the proposed approach, the writing result is evaluated by a wavelet transformation-based evaluation module, enabling the robot to learn to write without human supervision.

(2) Robot autonomy performance. Robots in the most of existing methods merely follow training data supplied by human engineers; therefore, these robots can be regarded as actuators with very low autonomy. In contrast, the proposed method exhibits several human-like patterns discussed in the next section, under the constraints of the LCAS algorithm.

(3) Control mode. All of these compared methods use an open-loop model, which requires human involvement in both the training and working phases. As a result, it increases the workloads of engineers. In contrast, the proposed method requires the engineer only to set the parameters of the constraint lifting sequence: subsequently, the robot learns by following this constraint sequence.

(4) Computational cost. The deep neural networks-based method must own a training phase to recognize or generate character images. In addition, the methods of learning writing from human gestures are based on real-time visual information. All of these settings need large amounts of computational resources. The main computational cost of the proposed methods is on the EP algorithm. Fig. 17 shows that the LCAS algorithm-based method requires fewer iterations to converge than does the regular EP; its computational cost is lower than these compared methods.

TABLE V
SUMMARY OF THE QUALITATIVELY COMPARISON WITH THE PREVIOUS APPROACHES.

Aspects:	Evaluation mechanism	Robot autonomy	Control mode	Computational cost
Method 1 [5]	Humans	Low	Open-loop	Medium
Method 2 [38]	Humans	Low	Open-loop	High
Method 3 [39]	Humans	High	Open-loop	High
Method 4 [40]	Minimization Differences	Medium	Open-loop	High
Proposed Method	Evaluation Algorithm	High	Closed-loop	Low

E. Discussion

In Figs. 8, 10, 12, 14, and 16, we noticed an important phenomenon: after constraint replacement, the evaluation scores decreased sharply. For example, in Fig. 10, at the fifteenth and twenty-seventh generations, the evaluation scores are significantly reduced. In fact, at the thirteenth and twenty-sixth generations, the stroke is in a saturated state; consequently, the constraints are replaced. This phenomenon, the (inverted) U-shaped phenomenon, exists mainly during infant development [31]. The (inverted) U-shaped phenomenon consists of a low error rate at the beginning of learning, followed by an unexpected increase in errors, and subsequently followed by improved performance and a return to a low error rate. This phenomenon has been extensively studied in psychology and has caused heated debates between the proponents of a rule-based strategy for syntax processing and the advocates of a distributed representation strategy [41], [42]. U-shaped learning phenomena have also been reported in other domains, including phonetic perception, face imitation, and explanations of a child's performance and errors due to changing representational strategies [31].

Many developmental robotics studies have modeled the progression of stages during robot development [43]. For example, Nagai et al. [44] explicitly modeled human joint attention developmental stages; several models have also directly addressed the modeling of U-shaped phenomena, such as the Morse et al. model of error patterns in phonetic processing [45]. In addition, motor babbling phenomena [46] play a critical role in many models of motor-skill acquisition and may help explain the U-shaped pattern. The LCAS method used in this paper also appears to present (inverted) U-shaped phenomena. Because the development process progresses through three stages, the constraints are placed twice; therefore, throughout the entire evolutionary process, for each stroke, the U-shaped phenomenon occurs twice.

As shown in Fig. 17, for the EP method, the evaluation score of each stroke rises steadily. Eventually, a better writing quality is achieved. However, the improvement increases occur more slowly in EP than in LCAS. In other words, under the same number of evolutionary generations, the writing quality of the LCAS method is sufficient. However, the EP method still needs to continue the evolutionary process—that is the EP method requires more evolutionary generations. Therefore, it is seen that the evolutionary method with LCAS significantly reduces the difficulty of the robot's learning.

These features of the proposed method are applicable to other robotic tasks; in particular, the robotic calligraphy writing is very similar to the robotic assembly task. Both of the two tasks require robots to invoke basic actions to form more complicated actions, so as to solve more difficult problems [47]. In addition, with the assistance of the LCAS algorithm, a complicated solution can be divided into a series of stages; thus, the proposed framework can also accelerate the solution process.

V. Conclusion

This paper proposed a developmental evolutionary learning framework for robots learning to write Chinese strokes. By combining the conventional evolutionary algorithm and a developmental learning mechanism, the robot system automatically learns how to write strokes through the framework. The entire learning process is divided into three developmental stages, in each of which different learning parameters are set. The robot iterates in each developmental stage until its learning status becomes stable; then, it moves to the next stage, and the procedure is repeated. Five common Chinese character strokes were used for this experiment. The results show that under our proposed framework, the robot learns stroke writing ability at a faster pace.

While our proposed approach shows promise, there is still room for improvement. First, the current evaluation model cannot handle the full diversity of written strokes. In addition, better aesthetic evaluation methods could be adopted to replace the current wavelet evaluation method to further improve the writing quality [1]. Likewise, better heuristic search algorithms, such as the differential evolution algorithm, harmony search algorithm, or particle swarm optimization algorithm, could be added to the framework to achieve a more powerful optimization performance. Another important factor in Chinese writing is to control the force of the pen. However, limited by the hardware of the robot and the material of the brush, the writing height was used to simulate the writing force and the height was set to a fixed value; therefore, this setting also constrains our robot's writing quality. In the future, we will consider adding more sensors to involve the writing force issue [48]. In addition, many parameters were empirically determined by the learning difficulty of each stroke in the current work. However, we believe that applying the Bayesian Information Criterion [49] to determine the developmental model's parameters, and using the expectation-maximization method to estimate means and variances can further improve the robotic autonomy performance.

References

[1] Z. Ma and J. Su, "Aesthetics evaluation for robotic Chinese calligraphy," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 1, pp. 80–90, March 2017.

[2] H. Zeng, Y. Huang, F. Chao, and C. Zhou, "Survey of robotic calligraphy research," *CAAI Transactions on Intelligent Systems*, vol. 11, no. 1, pp. 15–26, 2016.

[3] D.-t. Liang, D. Liang, S.-m. Xing, P. Li, and X.-c. Wu, "A robot calligraphy writing method based on style transferring algorithm and similarity evaluation," *Intelligent Service Robotics*, vol. 13, no. 1, pp. 137–146, 2020.

[4] Z. Ma and J. Su, "Stroke reasoning for robotic Chinese calligraphy based on complete feature sets," *International Journal of Social Robotics*, vol. 9, no. 4, pp. 525–535, 2017.

[5] F. Chao, Y. Huang, C. Lin, L. Yang, H. Hu, and C. Zhou, "Use of automatic Chinese character decomposition and human gestures for Chinese calligraphy robots," *IEEE Transactions on Human-Machine Systems*, vol. 49, no. 1, pp. 47–58, Feb 2019.

[6] N. García, J. Rosell, and R. Suárez, "Motion planning by demonstration with human-likeness evaluation for dual-arm robots," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–10, 2018.

[7] F. Xiong, B. Sun, X. Yang, H. Qiao, K. Huang, A. Hussain, and Z. Liu, "Guided policy search for sequential multitask learning," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 1, pp. 216–226, Jan 2019.

[8] K. Sasaki, K. Noda, and T. Ogata, "Visual motor integration of robot's drawing behavior using recurrent neural network," *Robotics and Autonomous Systems*, vol. 86, pp. 184–195, 2016.

[9] D. Berio, S. Calinon, and F. F. Leymarie, "Dynamic graffiti stylisation with stochastic optimal control," in *Proceedings of the 4th International Conference on Movement Computing*. ACM, 2017, pp. 18:1–18:8.

[10] Y. Sun, H. Qian, and Y. Xu, "Robot learns Chinese calligraphy from demonstrations," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2014, pp. 4408–4413.

[11] F. Chao, F. Chen, Y. Shen, W. He, Y. Sun, Z. Wang, C. Zhou, and M. Jiang, "Robotic free writing of Chinese characters via human robot interactions," *International Journal of Humanoid Robotics*, vol. 11, no. 1, pp. 1 450 007–1–26, March 2014.

[12] V. Mohan, P. Morasso, J. Zenzeri, G. Metta, V. S. Chakravarthy, and G. Sandini, "Teaching a humanoid robot to draw "shapes"," *Autonomous Robots*, vol. 31, no. 1, pp. 21–53, 2011.

[13] P. Liang, C. Yang, Z. Li, and R. Li, "Writing skills transfer from human to robot using stiffness extracted from sEMG," in *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, June 2015, pp. 19–24.

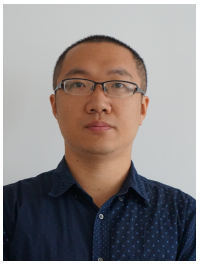
[14] C. Yang, S. Chang, P. Liang, Z. Li, and C. Y. Su, "Teleoperated robot writing using emg signals," in *2015 IEEE International Conference on Information and Automation*, Aug 2015, pp. 2264–2269.

[15] Y. Man, C. Bian, H. Zhao, C. Xu, and S. Ren, "A kind of calligraphy robot," in *IEEE International Conference on Information Sciences and Interaction Sciences*, China, 2010, pp. 635–638.

[16] X. Ma, Q. Kong, W. Ma, and X. Zhang, "4-DOF lettering robot's trajectory planning," in *Mechanical Engineering and Automation*, 2010, vol. 165, no. 5, pp. 161–163.

[17] S. Mueller, N. Huebel, M. Waibel, and R. D'Andrea, "Robotic calligraphy - Learning how to write single strokes of Chinese and Japanese characters," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013, pp. 1734–1739.
[18] H. I. Lin and Y. C. Huang, "Visual matching of stroke order in robotic calligraphy," in *2015 International Conference on Advanced Robotics (ICAR)*, July 2015, pp. 459–464.
[19] A. A. Saputra, J. Botzheim, and N. Kubota, "Evolving a sensory–motor interconnection structure for adaptive biped robot locomotion," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 11, no. 2, pp. 244–256, 2019.
[20] Z. Xie and Y. Jin, "An extended reinforcement learning framework to model cognitive development with enactive pattern representation," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 3, pp. 738–750, 2018.
[21] C.-F. Juang and T. B. Bui, "Reinforcement neural fuzzy surrogate-assisted multiobjective evolutionary fuzzy systems with robot learning control application," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 3, pp. 434–446, 2020.
[22] R. Wu, C. Zhou, F. Chao, L. Yang, C.-M. Lin, and C. Shang, "Ganccrobot: Generative adversarial nets based Chinese calligraphy robot," *Information Sciences*, vol. 516, pp. 474–490, 2020.
[23] R. Wu, C. Zhou, F. Chao, L. Yang, C.-M. Lin, and C. Shang, "Integration of an actor-critic model and generative adversarial networks for a Chinese calligraphy robot," *Neurocomputing*, vol. 388, pp. 12–23, 2020.
[24] M. H. Lee, Q. Meng, and F. Chao, "Staged competence learning in developmental robotics," *Adaptive Behavior*, vol. 15, no. 3, pp. 241–255, 2007.
[25] K. B. Lee and J. H. Kim, "Multiobjective particle swarm optimization with preference-based sort and its application to path following footstep optimization for humanoid robots," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 6, pp. 755–766, Dec 2013.
[26] S. Koos, J. B. Mouret, and S. Doncieux, "The transferability approach: Crossing the reality gap in evolutionary robotics," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 1, pp. 122–145, Feb 2013.
[27] C. F. Juang, M. G. Lai, and W. T. Zeng, "Evolutionary fuzzy control and navigation for two wheeled robots cooperatively carrying an object in unknown environments," *IEEE Transactions on Cybernetics*, vol. 45, no. 9, pp. 1731–1743, Sept 2015.
[28] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, and C. Yoshida, "Cognitive developmental robotics: A survey," *IEEE Transactions on Autonomous Mental Development*, vol. 1, no. 1, pp. 12–34, May 2009.
[29] S. Lemaignan, M. Warnier, E. A. Sisbot, A. Clodic, and R. Alami, "Artificial cognition for social human–robot interaction: An implementation," *Artificial Intelligence*, vol. 247, pp. 45 – 69, 2017, special Issue on AI and Robotics.
[30] Y. Jin and M. Lee, "Enhancing binocular depth estimation based on proactive perception and action cyclic learning for an autonomous developmental robot," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 1, pp. 169–180, Jan 2019.
[31] A. Cangelosi, M. Schlesinger, and L. B. Smith, *Developmental robotics: From babies to robots*. MIT Press, 2015.
[32] A. Giagkos, D. Lewkowicz, P. Shaw, S. Kumar, M. Lee, and Q. Shen, "Perception of localized features during robotic sensorimotor development," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 2, pp. 127–140, June 2017.
[33] R. Wu, C. Zhou, F. Chao, Z. Zhu, C.-M. Lin, and L. Yang, "A developmental learning approach of mobile manipulator via playing," *Frontiers in Neurorobotics*, vol. 11, p. 53, 2017. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnbot.2017.00053
[34] Xin Yao, Yong Liu, and Guangming Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, July 1999.
[35] A. P. Engelbrecht, *Computational Intelligence: An Introduction*, 2nd ed. Wiley Publishing, 2007.
[36] D. Wang, H. Liang, and Y. Yan, "An evaluation system for handwriting exercise of the Chinese character based on the characteristic extraction," *Techniques of Automation and Applications*, vol. 7, 2009.
[37] D. Zhou, J. Ge, R. Wu, F. Chao, L. Yang, and C. Zhou, "A computational evaluation system of Chinese calligraphy via extended possibility-probability distribution method," in *2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, July 2017, pp. 884–889.
[38] F. Chao, Y. Huang, X. Zhang, C. Shang, L. Yang, C. Zhou, H. Hu, and C.-M. Lin, "A robot calligraphy system: From simple to complex writing by human gestures," *Engineering Applications of Artificial Intelligence*, vol. 59, pp. 1–14, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0952197616302329
[39] D.-t. Liang, D. Liang, S.-m. Xing, P. Li, and X.-c. Wu, "A robot calligraphy writing method based on style transferring algorithm and similarity evaluation," *Intelligent Service Robotics*, vol. 13, no. 1, pp. 137–146, 2020.
[40] X. Zhang, Y. Li, Z. Zhang, K. Konno, and S. Hu, "Intelligent Chinese calligraphy beautification from handwritten characters for robotic writing," *The Visual Computer*, vol. 35, no. 6, pp. 1193–1205, 2019.
[41] S. Pinker and A. Prince, "On language and connectionism: Analysis of a parallel distributed processing model of language acquisition," *Cognition*, vol. 28, no. 1, pp. 73 – 193, 1988. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0010027788900327
[42] K. Plunkett and V. Marchman, "U-shaped learning and frequency effects in a multi-layered perception: Implications for child language acquisition," *Cognition*, vol. 38, no. 1, pp. 43 – 102, 1991. [Online]. Available: http://www.sciencedirect.com/science/article/pii/001002779190022V
[43] E. Ugur, Y. Nagai, E. Sahin, and E. Oztop, "Staged development of robot skills: Behavior formation, affordance learning and imitation with motionese," *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 2, pp. 119–139, June 2015.
[44] Y. Nagai, K. Hosoda, A. Morita, and M. Asada, "A constructive model for the development of joint attention," *Connection Science*, vol. 15, no. 4, pp. 211–229, 2003.
[45] A. Morse, T. Belpaeme, A. Cangelosi, and C. Floccia, "Modeling u shaped performance curves in ongoing development," in *Expanding the Space of Cognitive Science: Proceedings of the 23rd Annual Meeting of the Cognitive Science Society*, vol. 33, no. 33, 2011, pp. 3034–3039.
[46] D. Caligiore, T. Ferrauto, D. Parisi, N. Accornero, M. Capozza, and G. Baldassarre, "Using motor babbling and hebb rules for modeling the development of reaching with obstacles and grasping," in *International Conference on Cognitive Systems*, 2008, pp. E1–8.
[47] L. Gan, Z. Guo, F. Chao, L. Yang, X. Chang, C.-M. Lin, C. Zhou, V. Vijayakumar, and C. Shang, "Automatic stroke generation for style-oriented robotic Chinese calligraphy," *Future Generation Computer Systems*, vol. 119, pp. 20–30, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X2100039X
[48] X. Zhang, C. Zhou, F. Chao, C.-M. Lin, L. Yang, C. Shang, and Q. Shen, "Low-cost imu calibration with nonlinear scale factors," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2021.
[49] N. Dridi and M. Hadzagic, "Akaike and bayesian information criteria for hidden markov models," *IEEE Signal Processing Letters*, vol. 26, no. 2, pp. 302–306, 2019.

Ruiqi Wu received the BSc degree in computer science from Anyang Normal University, Anyang, China, in 2013, and the MAE degree in computer science from Henan Normal University, Xinxiang, China, in 2015. Then, he obtained the Ph.D. degree in computer science from Xiamen University in 2020. He is currently a Lecturer with the Department of Artificial Intelligence, School of Artificial Intelligence and Big Data, Henan University of Technology. His research interests are in Intelligent robots and machine learning. He has published more than 10 peer-reviewed journal and conference papers. His research interests are in adversarial learning, reinforcement learning, and Intelligent robots.



Fei Chao (M'11) received the B.Sc. degree in Mechanical Engineering from the Fuzhou University, China, and the M.Sc. Degree with distinction in Computer Science from the University of Wales, Aberystwyth, U.K., in 2004 and 2005, respectively, and the Ph.D. degree in robotics from the Aberystwyth University, Wales, U.K. in 2009. He is currently an Associate Professor with the School of Informatics, Xiamen University, China. Dr Chao has published more than 100 peer-reviewed journal and conference papers. His research interests include developmental robotics, machine learning, and optimization algorithms.

Changle Zhou received his PhD from Peking University in 1990. Currently, he is a professor of the Artificial Intelligence Department at the Xiamen University. He is also an affiliated professor of linguistics and applied linguistics in the Humanity College at the Zhejiang University, and an affiliated professor of the Philosophy Department at the Xiamen University. His research interests lie in the areas of artificial intelligence. His scientific contribution to the AI has more to do with machine consciousness and the logic of mental self-reflection. Beyond AI project, he also carries out research on a host of other topics including computational brain modeling, computational modeling of analogy and metaphor and creativity, computational musicology and information processing of data regarding traditional Chinese medicine. His philosophical works lie in ancient oriental thoughts of Chinese, such as ZEN, TAO, YI etc., viewed from science.

Yuxuan Huang received his B.Sc. and MEng degrees in Cognitive Science and Technology from the Xiamen University, China, in 2014 and 2017, respectively. He currently works as senior software engineer with the Software Development Center, Bank of Communications, China. His research focuses on optimization algorithms.

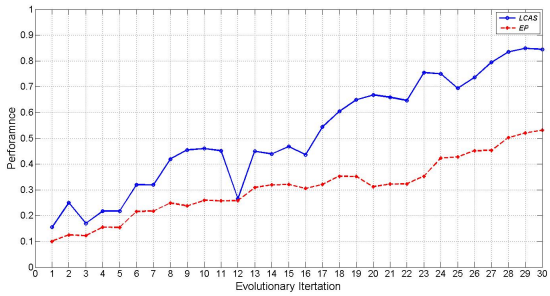
Longzhi Yang (M'12-SM'17) is currently a Programme Leader and a Reader with Northumbria University, Newcastle upon Tyne, U.K. His research interests include computational intelligence, machine learning, big data, computer vision, intelligent control systems, and the application of such techniques in real-world uncertain environments. He is the Founding Chair of the IEEE Special Interest Group on Big Data for Cyber Security and Privacy. Dr. Yang received the Best Student Paper Award from the 2010 IEEE International Conference on Fuzzy Systems.

Chih-Min Lin (M'87-SM'99-F'10) was born in Taiwan, in 1959. He received the B.S. and M.S. degrees from Department of Control Engineering and the Ph.D. degree from Institute of Electronics Engineering, National Chiao Tung University, Hsinchu, Taiwan, in 1981, 1983 and 1986, respectively. He is currently a Chair Professor and the Vice President of Yuan Ze University, Chung-Li, Taiwan. His current research interests include fuzzy neural network, cerebellar model articulation controller, intelligent control systems and signal processing. He has published more than 170 journal papers. He also serves as an Associate Editor of IEEE Transactions on Cybernetics and IEEE Transactions on Fuzzy Systems. He is an IEEE Fellow.

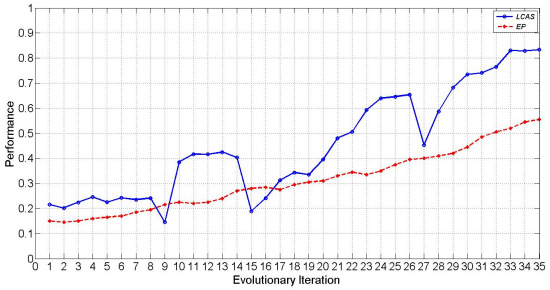
Xiang Chang received his BEng. degree in Cognitive Science at the Department of Artificial Intelligence, Xiamen University in 2019. Currently, he is a junior PhD student at the Department of Computer Science, Aberystwyth University. His research interests include deep reinforcement learning based robotic motion planning.

Changjing Shang received a Ph.D. in computing and electrical engineering from Heriot-Watt University, UK. She is a University Research Fellow with the Department of Computer Science, Institute of Mathematics, Physics and Computer Science at Aberystwyth University, UK. Prior to joining Aberystwyth, she worked for Heriot-Watt, Loughborough and Glasgow Universities. Her research interests include pattern recognition, data mining and analysis, space robotics, and image modelling and classification.

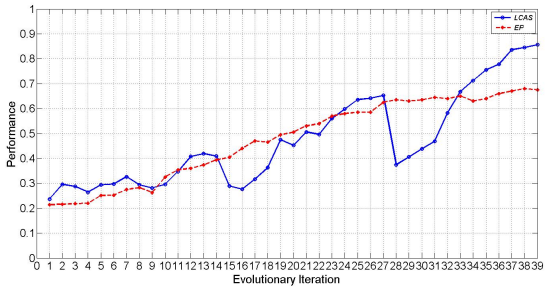
Qiang Shen received the Ph.D. in Computing and Electrical Engineering (1990) from Heriot-Watt University, Edinburgh, U.K., and the D.Sc. in Computational Intelligence (2013) from Aberystwyth University, Aberystwyth, U.K. He holds the Established Chair in Computer Science and is the Pro Vice-Chancellor: Faculty of Business and Physical Sciences, Aberystwyth University. His research interests include computational intelligence and its application in robotics. He has authored two research monographs and over 400 peer-reviewed papers, including one receiving an Outstanding Transactions Paper Award from the IEEE.



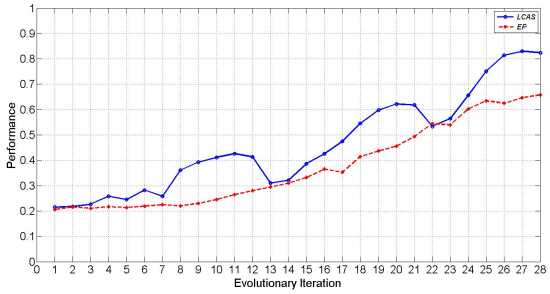
(a) Stroke 1



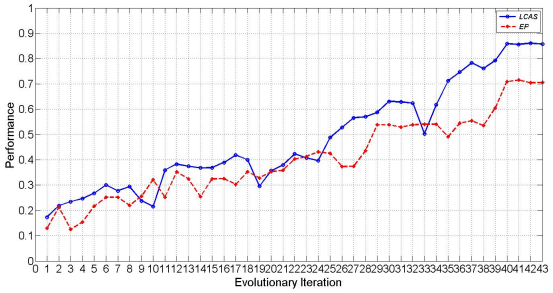
(b) Stroke 2



(c) Stroke 3



(d) Stroke 4



(e) Stroke 5

Fig. 17. The evolutionary method with development (LCAS method, solid curve), and evolutionary programming method without development (EP method, dotted line curve) in the autonomous generation process.